A Framework of Loose Travelling Companion Discovery from Human Trajectories

Elahe Naserian, *Member, IEEE*, Xinheng Wang, *Senior Member, IEEE*, Xiaolong Xu, *Member, IEEE*, and Yuning Dong, *Member, IEEE*

Abstract—Through the availability of location-acquisition devices, huge volumes of spatio-temporal data recording the movement of people is provided. Discovery of the group of people who travel together can provide valuable knowledge to a variety of critical applications. Existing studies on this topic mainly focus on the movement of vehicles or animals with forcing the group members to stay always connected. However, the movement of people is different; people might belong to the same main group while they contribute in various sub-groups during their movement. In this paper, we propose a group pattern called loose travelling companion pattern (LTCP), which allows the members of a group to contribute to various sub-groups as long as the community of members does not change during the movement and all of the members stay connected for a few time-slots. In addition, we propose weakly continuous loose travelling companion pattern (WCLTCP) to relax the continuous time constraint in LTCP. Finally, three algorithms have been developed to discover the proposed group patterns: (i) straightforward approach, (ii) smart-and-fast method, (iii) and opportunistic algorithm. Through the extensive experimental evaluation on both real and experimental datasets, the efficiency and effectiveness of the proposed group discovery approaches are proven.

Index Terms—movement trajectory, group pattern discovery, spatio-temporal data mining

1 Introduction

With the development of location-acquisition devices and tracking technologies, an enormous amount of streaming trajectory data recording the movement of people is available. Such data enables us to discover valuable knowledge about movement behavior of people. One of the interesting directions in this field is the discovery of group of people who move together because the people's group affiliations and their behaviors are highly correlated. For instance, by knowing the group(s) a person belongs to, retailers can derive common buying interests, develop group-specific pricing models, and provide personalized services.

Several studies have been proposed in the literature to discover the group patterns of moving objects [1], [2], [3], [4], such as flocks [5], [6], convoys [7], [8], swarms [9], gathering [10], [11], and Traveling Companion [12], [13]. The common thing among these patterns is that they all require the group to contain the same set of individuals during its lifetime. There exists another definition of group patterns, which doesn't need to put a strict requirement on members staying connected during the group's lifetime, like Moving Clusters [14], [15], [16], Evolving Convoy [17], or Loose Group Movement Pattern [18]. These group discovery methods are mainly focused on the movement of vehicles or animals with the aim of finding the general trends. For instance, convoy discovery (or traveling companion) can be applied for throughput planning of trucks or carpooling of vehicles; the discovery of common routes among commuters may be used for scheduling of collective transport; and the moving cluster discovery (or Loose Group Movement Pattern) can be used for animal studies or traffic analysis, i.e., the discovery of the pathways of species migration or the identification of traffic in busy areas.

Different from previous research work, we concentrate on human movement trajectories, particularly in the indoor environment. The goal is to discover groups of people to improve the provided services to them. However, the movement of people is rather different from the movement of animals or vehicles. People might belong to one group, although they have different movement paths (unlike the convoy or traveling companion). Various sub-groups may be also formed during their lifetime. Fig.1 represents an example of group movement of passengers waiting at the airport before their departure (according to the observation of movement data of passengers at Guangzhou Baiyun International Airport). As it is shown in the figure, while these passengers belong to the same main group, they also contribute in different sub-groups. There are plenty of other examples that correspond to this kind of movement pattern, such as a group of tourists visiting a museum, or a group of friends browsing in a shopping mall.

Discovering this kind of group relationships (sub-groups in addition to the main group) brings the following advantages: firstly, the discovery of sub-groups provides more detailed information of the users for the authorities. Taking the airport as an example, with having access to the profile information of the passengers (which is available

- *E. Naserian is with the School of Engineering and Computing, University of the West of Scotland, United Kingdom. E-mail: elahe.naserian@uws.ac.uk*
- *X. Wang is with the School of Computing and Engineering, University of West London, United Kingdom. E-mail: xinheng.wang@uwl.ac.uk*
- *X. Xu is with the Jiangsu Key Laboratory of Big Data Security and Intelligent Processing, Y. Dong is with the College of Telecommunications and Information Engineering, Nanjing University of Posts and Telecommunications, China. E-mail: (xuxl, dongyn)@njupt.edu.cn*

to the airport authority), the type of the groups can be identified. The group in Fig.1 could be a group of friends, as the main group, which is composed of a family and a couple, as the sub-groups. Through this classification, airport authority can provide more accurate personalized services. For example, the services that can be recommended to a friends group comprised of a family might be different if we only take the friends group into account. Secondly, letting the members contribute in different sub-groups leads to the discovery of complete and long-term groups. In the previous example, one group including the main group as well as the sub-groups will be discovered which covers the whole movement, rather than the discovery of multiple fragmented and partial groups.

However, designing a system capable of discovering such group patterns has the following challenges:

- Non-strict continuity: Groups are composed of people who travel together, but not necessarily always together. It's much clearer in the case of large groups, which contain the sub-groups with independent movements. Many state-of-the-art group discovery methods, retrieving the groups whose members are always connected, ignore the existence of sub-groups [7], [8], [9].
- High Precision: In some applications, like providing the personalized services, the wrong group discovery has a worse effect than not discovering the group. Contrary to the state-of-the-art methods which are focusing on discovering more groups, our focus is on the discovery of the correct groups.
- Incremental discovery: The groups need to be discovered as soon as possible in order to provide the real-time services. Therefore, the groups should be reported in an incremental manner such that the discovery algorithm has to output the results while processing the trajectory data stream, simultaneously.
- Efficiency: As trajectories are generated in a format of the data stream, we are dealing with huge amounts of data arriving in a short period of time. Hence, the discovery algorithm should be able to output the groups in an efficient manner.
- Effectiveness: As the number of the groups can be quite large, the group discovery model should be able to output the complete and long-term groups rather than partial and short-term ones in an effective manner.

In this paper, we propose the notion of Loose Travelling Companion Pattern (LTCP) framework, which is a sequence of cluster-sets, such that the cluster-sets contain the same objects and all of the members have to stay in the same cluster in the predefined number of time-slots. Accordingly, we propose three algorithms to identify LTCPs in a spatio-temporal dataset. The first algorithm is a straightforward method which directly follows the problem definition. The second algorithm speeds up the process of candidate extension through a smart and fast strategy. Finally, an opportunistic approach is proposed for improving the new candidate creation process. We also extend the proposed methods to adapt to some complicated scenarios. In case of lots of objects in a limited space, i.e., airport, there exist

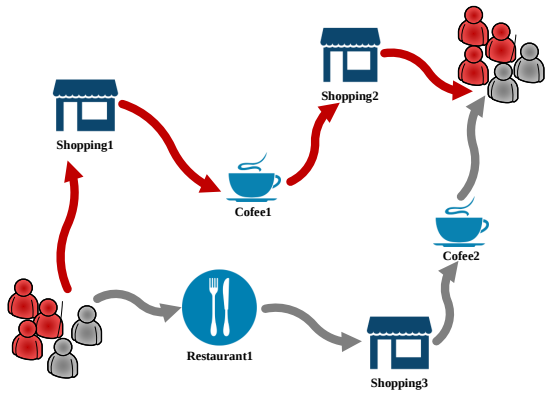


Fig. 1. A group of passengers at airport

some moments that the objects in a group may temporarily stay in the same cluster with other groups. Taking this into account, we propose a Weakly Continuous Loose Travelling Companion Pattern (WCLTCP) which relaxes the consecutive time constraint in the LTCP.

This paper substantially extends the conference version [19] in the following ways: (1) proposing a smart and fast approach to avoid the redundant checks and accelerate the candidate extension in each time-slot; 2) identifying the bottleneck of the problem and proposing an efficient approach to create the new candidates from the current clusters; 3) expanding experimental studies on performance of the proposed methods, and also evaluating the quality of the discovered groups.

The remainder of this paper is organized as follows. The related work is discussed in Section II. The proposed model is introduced in Section III. In Section IV, the algorithms for discovering LTCP patterns are presented in detail. Section V provides the definition of WCLTCP. The experimental study is presented in Section VI, and Section VII concludes this paper.

2 Related Works

The problem of pattern discovery from the spatio-temporal trajectories has been variously formulated in the literature. As surveyed in [20] there are four major categories of patterns that can be identified from the trajectories. The first one is the Trajectory Clustering, which aims to discover a representative path by grouping the similar trajectories into clusters [21], [22], [23], [24]. The second one is the Frequent Sequential Pattern Mining, which tries to discover the common sequence of locations in a similar time interval [25], [26], [27], [28]. Periodic pattern mining is the third category which is based on the periodic nature of activity patterns [29], [30], [31], [32]. The fourth category, which is the focus of our paper, aims to discover the group of objects which move together, Group pattern mining. We distinguish related work in this area based on whether the membership is constant during the lifetime of the group or not.

Flock pattern is one of the earliest work proposed in this area [5]. In this pattern, the group members move together within a constant circular region for k consecutive time-slots. Even though several variants of this model have been

proposed in the literature [6], [33], [34], they all have the limitation of moving inside a circle with a predefined radius. In [7], Jeung stepped forward and proposed a framework of convoy pattern which allows a pattern of any shape and extent. A convoy is defined as a group of objects which are connected to each other for at least k consecutive time intervals. The traveling companion proposed by Tang [12] is essentially consistent with the convoy pattern, and the main contribution of this work is to accelerate the pattern discovery algorithm. While all of these patterns have some strict requirements on the consecutive time period, a more general pattern is proposed by Li et al [9] as Swarm. In this pattern, objects should stay close together for at least k time intervals, which could be non-consecutive. The definition of the group pattern in [35] is a mixture of the flock and swarm patterns. Group patterns are the moving objects that travel within a fixed radius for at least k time intervals, which could be non-consecutive. Even though the relaxation issue has been addressed, same as swarm, size and shape of the group has been restricted to a predefined radius, same as the flock. This group pattern also results in the redundant group discovery which makes the algorithm exponentially inefficient. A similar idea is used in finding the loose companion [13], which relaxes the continuous requirement in [12] through letting the members not to be connected for a predefined time intervals.

The major difference of the mentioned work is about the discovery algorithm. Most of them are designed to work on the static datasets and are not able to output the results in an incremental manner. The convoy algorithm is one of them, which needs to load the whole trajectory data into memory. The discovery algorithm of the swarm patterns also needs to load the whole dataset into memory. Accordingly, these approaches are not applicable in a data streaming environment. Travelling companion, on the other hand, applies an incremental manner which is suitable for the streaming data, similar to our approach. Despite the slight differences, all the mentioned patterns have the same requirement that members need to stay connected for the whole lifetime of the group (consecutive or non-consecutive). In our proposed pattern, however, members are able to be disconnected from the group and have their own movement. Therefore, the above mentioned methods are not applicable to model our proposed group pattern. A new group pattern is required to describe this kind of group movement, which is quite common nowadays in large indoor commercial and service environments.

Following illustrative examples are used to demonstrate the limitations of previously proposed group patterns.

Example 1: Fig. 2 shows a group pattern with nine objects $\{O_1, O_2, O_3, O_4, O_5, O_6, O_7, O_8, O_9\}$. The members belong to the same main group while they follow different paths and form various sub-groups ($\{O_1, O_2, O_3, O_4, O_5\}$, $\{O_6, O_7\}$, and $\{O_8, O_9\}$). Considering the size requirement of 2 ($m_G = 2$), and duration threshold of 5 time-slots ($d_G = 5$), the following convoys can be discovered: $\{O_1, O_2, O_3, O_4, O_5\}$, $\{O_6, O_7\}$, and $\{O_8, O_9\}$. As it is evident, the patterns discovered by the convoy method are the sub-groups not the main group. We call this problem partial discovery. The reason is, in a convoy group pattern, members need to stay connected in one cluster during the whole group's lifetime, which makes it unable to apply for LTCP group. Swarm and loose companion follow the same approach. The only difference is relaxing the continuous requirement.

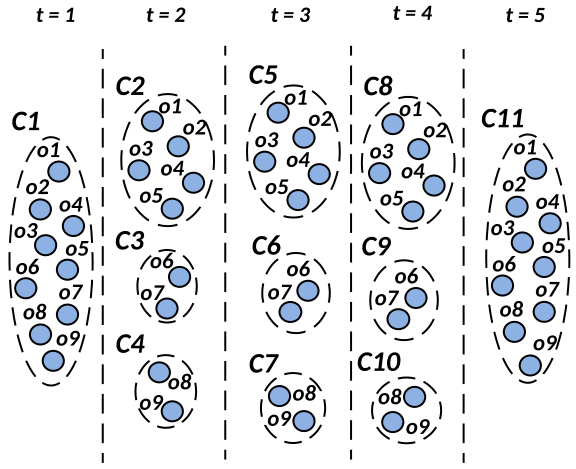


Fig. 2. Example of group movement

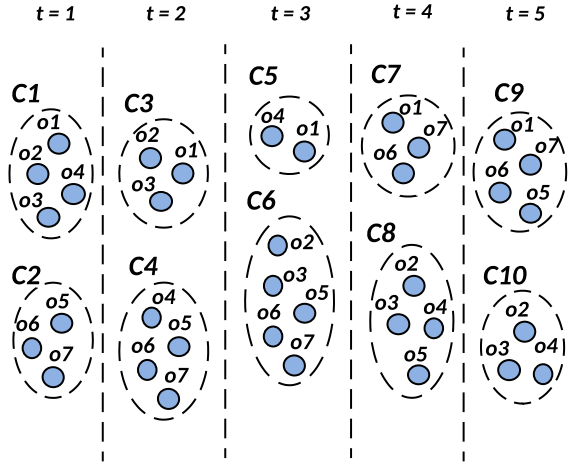


Fig. 3. Example of random movement

In another line of study, members are allowed to leave the group and new members to join the group during the lifetime of a group [14], [17], [18]. Kalnis proposed the notion of a moving cluster [14], which is a sequence of spatial clusters appearing during consecutive time-slots, such that the portion of common objects in any two consecutive clusters is not below a given threshold parameter θ. Let c_t and c_{t+1} be clusters at times t and $t+1$, respectively. These clusters belong to the same moving cluster if $|c_t \cap c_{t+1}| / |c_t \cup c_{t+1}| \geq \theta$, where θ is a user-specified threshold value between 0 and 1. It should be noted that a moving cluster pattern is strongly affected by the value of θ.

Example 2: In Fig. 2, if we set $\theta = 0.9$ (i.e., requiring 90% overlapping clusters), the overlap between c_1 and the clusters in time-slot 2 (c_2, c_3, and c_4) is up to 55% (between c_1 and c_2) and no pattern will be discovered as a moving cluster. On the other hand, if we set $\theta = 0.2$, the group $\{O_1, O_2, O_3, O_4, O_5, O_6, O_7, O_8, O_9\}$ will be dis-

covered three times with different cluster sequences which leads to the redundant discovery.

Example 3: Fig. 3 shows the random movement of 7 objects between time-slots 1 to 5. If we set $\theta = 0.5$, the wrong group $\{O_2, O_3, O_4, O_5, O_6, O_7\}$ with the cluster sequence $\langle c_2, c_4, c_8, c_{10}\rangle$ will be discovered as a moving cluster.

Even though in this movement pattern there is no strict requirement for the members to be always connected, it still suffers from the following problems: 1) Moving cluster doesn't necessarily maintain the same set of members during the lifetime of the group. As each cluster should share enough objects only with the consecutive cluster, the members of the last cluster could be totally different from the first cluster, example 3. 2) There is no certain θ value that can be used to identify the accurate LTCP groups, either the wrong groups may be discovered, example 3, or the real LTCPs may remain undiscovered, example 2, or the redundant groups might be discovered. 3) There is no lifetime constraint for this group pattern. A moving cluster can be formed as long as two consecutive clusters have enough overlap, even for only two consecutive time-slots.

The definition of Evolving convoy [17] is similar to the moving cluster, which relaxes the strictness of connecting the members all the time together. A $w - convoy$ is defined as w continuous clusters which contains the persistent members (those who are connected at w continuous timestamps) and dynamic members (those who are connected with the persistent members for at least k time intervals). A $w - convoy$, then, could be connected by the subsequent $w - convoy$, if they have $w - 1$ same timestamps and have certain common persistent members. It should be noted that if we set $w = 2$, the evolving convoy and moving cluster will be the same.

Weakly Consistent Group Movement Pattern (WCM) [18] is another group movement definition which has the most similarity to our work. According to this group definition, each w continuous clusters should contain at least m_C persistent members (those who are connected at w continuous timestamps), and also each member can leave the whole for l_C time intervals. Even though WCM doesn't retain the members to stay in one cluster during the group's lifetime, there is a significant difference between a WCM and an LTCP pattern.

Example 4: Considering Fig. 2, if we set $w = 2$ (size of window) with $m_C = 2$ (number of persistent members), and $l_C = 2$ (number of time intervals that a member can leave the group), the following WCMs can be discovered: $\{O_1, O_2, O_3, O_4, O_5\}$, $\{O_6, O_7\}$, and $\{O_8, O_9\}$. As it is clear the main group (which contains all the objects) cannot be discovered. The reason is the time period that members are disconnected is 3 time-slots which is more than the defined threshold, l_C. On the other hand, if we set $l_C = 3$, the main group will be discovered, but for 3 times.

Example 5: If we consider the same setting as above for Fig. 3, WCM results the wrong group $\{O_2, O_3, O_4, O_5, O_6, O_7\}$ from a random movement. This group is discovered while the members have never been gathered.

The following reasons make WCM unable to to discover the exact LTCP results: 1) Even though it allows members to be disconnected from the group, it restricts that through putting the limitation on the maximum time a member can leave the group. It makes WCM unable to discover the groups composed of sub-groups with independent movements, example 4. 2) On the other hand, this condition might lead to discovery of wrong groups whose members have never been gathered, example 5. This problem gets worse with increasing the allowed time-gap l_C, specially with larger groups. 3) It generates too many redundant groups. The reason is each candidate can be extended to more than one in each time-slot, even with the same members, which leads to the high redundancy of the group discovery results, example 4. More importantly, it substantially degrades the performance of the algorithm, because of the high space cost (and accordingly time cost), which gets worse with increasing the allowed time-gap. The above reasons have been proved in the experimental study.

Same as the second category (moving cluster and WCM), LTCP also doesn't need the members to stay together all the time. However, unlike the moving cluster or evolving convoy, the membership doesn't change during the group's lifetime. On the other hand, there is no limit on the period that members can leave the group, unlike the WCM, as long as they don't get mixed up with the other groups and also stay together for a certain time-slots. Through this approach, we are able to discover the sub-groups, which group members contribute during their movement, in addition to the main group. Finally, all of the previous works suffer from the limitation of considering just one cluster at each time-slot, which makes them impractical to model this kind of group pattern (a full exemplary discovery process of LTCP is outlined in Table 2). Here is a summary of the major differences of the state-of-the-art approaches and LTCP:

- First category of patterns, Convoy, Swarm, and Loose companion, put strict requirement on the groups to be gathered all the time, which leads to the partial group discovery of groups.
- There is no absolute θ value for Moving cluster

TABLE 1
Table of Notations

Notation	Definition
O	Moving object
Tr	Trajectory
t	time-slot index
O_{DB}	Set of objects
c	Cluster
$c.f$	Frequency of a cluster
C_i	Set of clusters at time-slot i, slot-cluster
C_{DB}	Set of slot-clusters at all time-slots
cs	Subset of C_i, cluster-set
S_i	Set of all subsets of C_i, subset-collection
$cs.t$	Timestamp of a cluster-set
P	Pattern
$P.O_G, P.lifetime$	Member set and lifetime of P
$P.TS$	Cluster-set sequence of P
m_G, d_G	Size and duration threshold
f_C	Frequency threshold of gathering of all members
l_C	Gap threshold between cluster-sets

pattern that can be used to compute the exact LTCP results, either false hits may be found, or actual LTCPs may remain undiscovered.
- In Moving Cluster and Evolving Convoy patterns, the membership is changing during the group's lifetime which leads to the wrong group discovery.
- In WCM, because of putting the limitation on the period that a member can leave the group, the actual LTCPs may remain undiscovered.
- In WCM, as there is no constraint on staying the whole group gathered, false groups may be found.

In a summary, none of the above approaches are able to discover the sub-groups that group members form during their movement, which is because of considering the single clusters at each time-slot.

3 Problem Definition

Definitions of all essential concepts used throughout the paper will be presented in this section. The list of main notations is outlined in Table 1.

$O_{DB} = \{O_1, O_2, ..., O_n\}$ is a set of moving objects. The trajectory Tr of a moving object O is represented as a series of points denoted as $Tr = \langle p_1, p_2, ..., p_n \rangle$, where p_i includes the location and timestamp attributes. We assume $t \in \{1, ..., T\}$ as the time interval, which T may be equal to a day or a time-slot, i.e., one minute, depending on the requirement of applications. The set of objects with their trajectories at time-slot t is called a **slot-dataset** at t.

Clustering method is considered as a pre-processing step and it depends on the characteristics of the data. As trajectories may have different lengths and sampling rates, we applied Longest Common Subsequence (LCSS) as a distance measure that allows stretching sequences over time. This measure allows objects that are close in space at different time instants be matched if the time instants are also close [36]. We applied hierarchical clustering, as it is compatible with LCSS as a non-metric similarity measure [36]. Unlike other clustering approaches, like k-means, hierarchical clustering takes no input parameters. It also results the full clustering, even clusters with small sizes, which makes it appropriate for our goal, finding the groups even with two members. It should be noted, while clustering method is not fixed in our framework, those who generate overlapping clusters are not applicable for our model.

The output of the clustering step is a database of **slot-clusters** $C_{DB} = \langle C_1, C_2, ..., C_n \rangle$. Each slot-cluster C_i contains the extracted clusters at time-slot i, $C_i = \langle c_{i,1}, c_{i,2}, ..., c_{i,j} \rangle$, where j is the number of clusters at that time-slot. The number of time-slots that a cluster c has been observed is denoted by $c.f$. We call the subset set of clusters at time-slot i, **subset-collection** $S_i = \langle cs_{i,1}, cs_{i,2}, ..., cs_{i,k} \rangle$, where k is the number of subsets (except empty subset). Each subset cs is called a **cluster-set**. For example, in Fig. 2, at time-slot 2, we have the slot-cluster $C_2 = \langle c_2, c_3, c_4 \rangle$ and subset-collection $S_2 = \langle \langle c_2 \rangle, \langle c_3 \rangle, \langle c_4 \rangle, \langle c_2, c_3 \rangle, \langle c_2, c_4 \rangle, \langle c_3, c_4 \rangle, \langle c_2, c_3, c_4 \rangle \rangle$, and the cluster-set $cs_{2,4} = \langle c_2, c_3 \rangle$. For easy readability, we refer to the timestamp of a cluster-se as $cs.t$.

An LTCP group is a sequence of cluster-sets at continuous time-slots. We define four threshold parameters in order to identify the LTCP groups: 1) size threshold m_G which restricts the size of the groups we are targeting, 2) duration threshold d_G which determines the minimum lifetime of a group, 3) frequency threshold f_C that determines the minimum time-slots that a group should be gathered.

$Definition\ 1$: According to the above mentioned parameters, we identify an object set O_G along with a cluster-set sequence $TS = \langle cs_{1,a}, cs_{2,a2}, ..., cs_{n,an} \rangle$ ($n = t_2 - t_1 + 1$) at interval $[t_1, t_2]$ as an LTCP group, if it satisfies the following conditions:

1) $cs_1.t = t_1$, $cs_n.t = t_2$, $|t_2 - t_1| \geq d_G$;
2) $|O_G| \geq m_G$;
3) For any $cs_{i,ai}$ of TS, union of its clusters should be equal to $O_G : \forall\ c_j \in cs_{i,ai}$, $\bigcup c_j = O_G$;
4) $\exists\ c_j \in TS$, $c_j = O_G$, and $c_j.f \geq f_C$;

then the pair of object set O_G and cluster-set sequence TS is defined as **Loose Travelling Companion Pattern (LTCP)** in $[t_1, t_2]$, $P = \langle O_G, TS \rangle$. Conditions 1 and 2 guarantee that lifetime and size of the group satisfy the duration and size threshold, respectively. Based on the union operation among clusters in a cluster-set, a sequence of cluster-sets is derived which meets condition 3. Condition 4 adds a constraint on the number of time-slots that all members of an LTCP group should stay close together. Fig. 2 shows an example of LTCP group with 9 members $\{O_1, O_2, O_3, O_4, O_5, O_6, O_7, O_8, O_9\}$ in interval [1, 5] while it satisfies the condition of $f_C = 2$ and $d_G = 5$.

We define an LTCP pattern such that the precise groups are able to be discovered while considering the nature of human movement behaviour, non-strict continuity. Accordingly, condition 3 provides the group members with the freedom to be disconnected and have independent movements, which enables us to discover the sub-groups and satisfies the non-strict continuity feature. On the other hand, members shouldn't stay with other groups (condition 3) during their movement and also all the group members have to meet each other at least for a certain time-slots (condition 4) which results the hight precision discovery. The condition 3 will also lead to more effective group discovery, resulting the complete and long-term groups rather than partial and short-term ones. As a result, LTCP outputs the lower number of groups compared to the state-of-the-art approaches, however, the quality of the discovered groups (precision and effectiveness) are remarkably higher. Experimental results in Section VI will prove this.

4 Discovery Algorithms of LTCP Patterns

In this section we present three algorithms for discovering LTCPs. The first algorithm is a straightforward implementation of the problem definition. The second one, a smart and fast discovery algorithm, improves the efficiency by avoiding the redundant checks. The final version is an opportunistic algorithm which proposes an efficient candidate creation method.

4.1 Straightforward Approach

Firstly, we introduce the straightforward approach to discover the LTCPs which directly follows the problem definition. This algorithm is described in detail by pseudocode in

Algorithm 1 : Straightforward Approach

Input: Tr_{DB}, m_G, d_G, f_C
Output: all closed LTCPs

1: $V := \emptyset$; $//set\ of\ current\ LTCP\ candidates$
2: **for** *each time slot* t **do**
3: $V_{next} := \emptyset$; $//new\ set\ of\ LTCP\ candidates$
4: cluster $O_{DB}(t)$ as C
5: **for** *each candidate* $P \in V$ **do**
6: $P.extended := false$;
7: $temp := \emptyset$;
8: **for** *each cluster* $c \in C$ **do**
9: **if** c *is a subset of* $P.O_G$**:**
10: add c to $temp$;
11: **if** $temp.O_G = P.O_G$**:**
12: $P.extended := true$;
13: $P_{next} := \langle P.O_G, P.TS\ o\ temp \rangle$;
14: add P_{next} to V_{next};
15: **if** $P.extended = true$**:**
16: **if** P is a valid $LTCP$
17: $P.valid = True$;
18: **else if** $P.valid = true$ and P is closed**:**
19: output P as a closed LTCP;
20: generate closed subset-collection (C, V_{next}, m_G) as S;
21: **for** *each* cluster-set cs *at* S **do**
22: $P_{next} := \langle cs.O_G, \langle \text{cs} \rangle \rangle$; $//initialize\ a\ new\ candidate$
23: add P_{next} *to* V_{next};
24: $V := V_{next}$

Algorithm 1. Algorithm 1 is composed of three parts: clustering (Line 4), the extension of current candidates (Lines 5-19), and new candidate creation (Lines 20-23). In the first part, we apply a hierarchical clustering on sub-trajectories of objects in the current time-slot and extract the clusters. The details of this part are eliminated since it is not the affirmation of our problem.

In the second part, this algorithm refines the candidates according to the discovered clusters (Lines 5-19). Considering the current candidate P, for each cluster c, if its members are a subset of the candidate's members, it will be added to a temporary list, $temp$ (Lines 7-10). After reviewing all the current clusters, if the temporary list and the current candidate have the same members, P_{next} will be derived from P (Lines 11-14). Once candidate P is extended in the current time-slot, it will be checked whether it is a valid LTCP or not according to the conditions in Definition 1 (Lines 15-17), which means lifetime of the candidate will be checked considering the duration threshold, and the number of times that all members are gathered will be checked according to the frequency threshold. Otherwise, it will be checked if it is a closed LTCP. If there does not exist any LTCP in the candidate list, which is a sup-pattern of P, then P is recognized as a closed LTCP (Lines 18-19).

$Definition\ 2$: Considering two LTCPs of $P = \langle O_G, TS \rangle$ and $P' = \langle O'_G, TS' \rangle$, if $O_G \subseteq O'_G$ and all cluster-sets in TS is subsets of cluster-sets in TS', P is a sub-pattern of P', and P' is a sup-pattern of P. An LTCP is said to be a **closed pattern**, if it has no sup-patterns.

The new candidate creation is the final step of LTCP discovery algorithm (Lines 20-23). In this part, the subset-collection, S, would be extracted from the current clusters. Only closed subsets, according to the candidate set V_{next}, which satisfy the size requirement m_G, would be added to the new candidate list (Line 23). For a subset cs_i, if there does not exist a candidate P_j such that $cs_i.O_G = P_j.O_G$, then cs_i is a **closed-subset**.

Considering all of the combinations of new clusters as possible candidates would cause the exponential complexity to the algorithm. In order to reduce the computation time and space of this step, we put a threshold for the maximum size of the groups according to the application. In the case of the airport application, we consider the maximum group size of 17. We also just allow the passengers who belong to the same flight create a group.

Example 6: Table 2 shows the running process of the LTCP discovery algorithm. It is clear that the membership is unchanged during the lifetime of the group ($O_G = \{O_1 - O_9\}$); however members are contributing in different sub-groups (clusters). As all members stay close together in two time-slots ($t = 1$ and 5), the fourth condition is also satisfied for $f_C = 2$. It is worth pointing out that the threshold parameters could vary according to the applications. For example, for airport case, as we are interested in discovering even groups of two, we set $m_G = 2$.

4.2 Smart-and-Fast Approach

The computational overhead of the straightforward approach could be high because of two major time-consuming operations: finding the qualified cluster-set which matches with the candidate, and also creation of the new candidates. In this subsection, we try to improve the efficiency of the first time-consuming operation, where the second problem will be resolved in next subsection by a proposed opportunistic approach. In each time-slot, every pair of candidate and cluster is checked to see if the cluster is a subset of the candidate. However, most of these subset checkings are unnecessary.

Lemma 1: let P be a candidate and c be a cluster in the current time-slot. If $c \cap P.O_G \neq \emptyset$, P is extendable if and only if $c \subseteq P.O_G$.

Proof: According to definition 1, a candidate is extendable if a cluster-set comprising all of its objects, and just its

TABLE 2
Illustration of LTCP discovery

time	cluster-set	candidates
1	C_1	$P_1 = \langle\ O_G, \langle\langle C_1 \rangle\rangle\ \rangle$
2	C_2,C_3,C_4	$P_2 = \langle\ O_G, \langle\langle C_1 \rangle, \langle C_2, C_3, C_4 \rangle\rangle\ \rangle$
3	C_5,C_6,C_7	$P_3 = \langle\ O_G, \langle\langle C_1 \rangle, \langle C_2, C_3, C_4 \rangle, \langle C_5, C_6, C_7 \rangle\rangle\ \rangle$
4	C_8,C_9,C_{10}	$P_4 = \langle\ O_G, \langle\langle C_1 \rangle, \langle C_2, C_3, C_4 \rangle, \langle C_5, C_6, C_7 \rangle, \langle C_8, C_9, C_{10} \rangle\rangle\ \rangle$
5	C_{11}	$P_5 = \langle\ O_G, \langle\langle C_1 \rangle, \langle C_2, C_3, C_4 \rangle, \langle C_5, C_6, C_7 \rangle, \langle C_8, C_9, C_{10} \rangle, \langle C_{11} \rangle\rangle\ \rangle$

Algorithm 2 : Smart-and-Fast Approach

Input: Tr_{DB}, m_G, d_G, f_C
Output: all closed LTCPs

1: $V := \emptyset$; $//set\ of\ current\ LTCP\ candidates$
2: **for** $each\ time\ slot\ t$ **do**
3: $V_{next} := \emptyset$; $//new\ set\ of\ LTCP\ candidates$
4: cluster $O_{DB}(t)$ as C
5: **for** $each\ candidate\ P \in V$ **do**
6: $P.extended := false$;
7: $temp := \emptyset$;
$***$
8: **while** $true$**:**
9: retrive a random object o of P;
10: extract cluster c which contains the object o;
11: **if** $intersection\ of\ c\ and\ P.O_G\ is\ not\ null$**:**
12: **if** $c\ is\ a\ subset\ of\ P.O_G$**:**
13: add c to $temp$;
14: **else:**
15: $break$;
16: **if** $temp.O_G = P.O_G$**:**
17: $P.extended := true$;
18: $P_{next} := \langle P.O_G, P.TS\ o\ temp\rangle$;
19: add P_{next} to V_{next};
20: $break$;
$***$
21: **if** $P.extended = true$**:**
22: **if** P is a valid $LTCP$
23: $P.valid = True$;
24: **else if** $P.valid = true$ and P is closed**:**
25: output P as a closed LTCP;
26: generate closed subset-collection (C, V_{next}, m_G) as S;
27: **for** $each$ cluster-set $cs\ at\ S$ **do**
28: $P_{next} := \langle cs.O_G, \langle$cs$\rangle\rangle$; $//initialize\ a\ new\ candidate$
29: add $P_{next}\ to\ V_{next}$;
30: $V := V_{next}$

objects, can be discovered. In other words, other objects are not allowed to stand in the same cluster as the candidate's object.

Through Lemma 1 the process of finding a cluster-set for a candidate will stop earlier if the other objects stand in the same cluster as the candidate's object (Line 15). This is because of the fact that the candidate's objects should remain unchanged during its lifetime. Therefore, a candidate cannot be extended to more than one candidate during the next time-slot. Accordingly, there is no need to do more comparisons when the temporary list matches with the candidate (Line 20).

Every candidate shares objects only with a few number of clusters, so most of the clusters don't have objects in common with the candidate. Therefore, we search for the clusters which contain the candidate's objects instead of checking the candidate with all of the clusters. We select a random object of candidate P and search for it in all clusters of C (Lines 9-10). If the retrieved cluster satisfies Lemma 1,

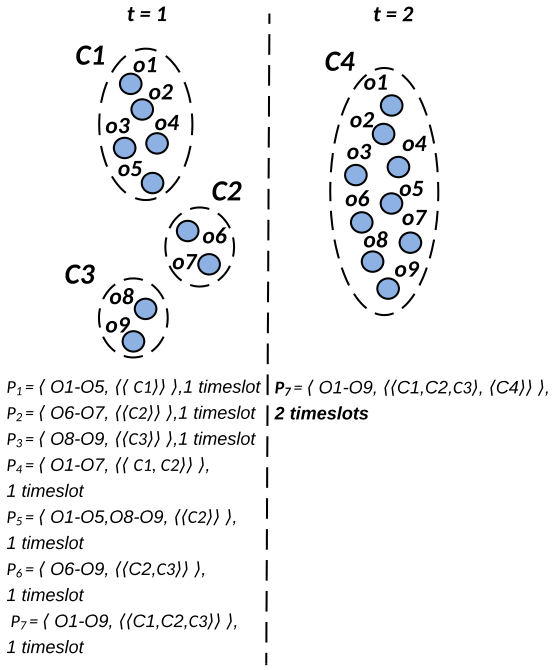


Fig. 4. Candidate creation problem

it will be added to the temporary list, otherwise, we go to the next candidate. We repeat this process with choosing another candidate's object, which is not observed yet, until the temporary list contains all of the candidate's objects or Lemma 1 is not satisfied.

Considering Lemma 1 and efficient search of clusters, we propose a smart-and-fast algorithm. Algorithm 1 is then modified and a new algorithm is presented in Algorithm 2. The modified part is marked with * in Algorithm 2. In order to implement an efficient search of an object in the clusters of C, we generate a hash table which contains all objects in the time-slot and their corresponding clusters. It leads to finding the object in constant time, on average.

4.3 Opportunistic Approach

Although the efficiency of the LTCP discovery algorithm is improved through the smart-and-fast approach, the system still suffers from the computational overhead of candidate creation step in both time and space. In each time-slot, all of the combinations of new clusters are considered as new candidates, which leads to the exponential complexity of the algorithm. In the straightforward algorithm we tried to reduce this complexity through putting the threshold on the size of the valid candidates, and also consider only the passengers who belong to the same flight as the possible groups. However, this is not a generally applicable approach. In this sub-section, we introduce an efficient candidate creation approach which decreases the number of generated candidates substantially.

According to definition 1, in order to recognize a candidate as an LTCP, all of the members have to stay in the same cluster for predefined time-slots, f_C. Therefore, most of the candidates generated in the new candidate creation step cannot be qualified as an LTCP. Fig. 4 shows two consecutive

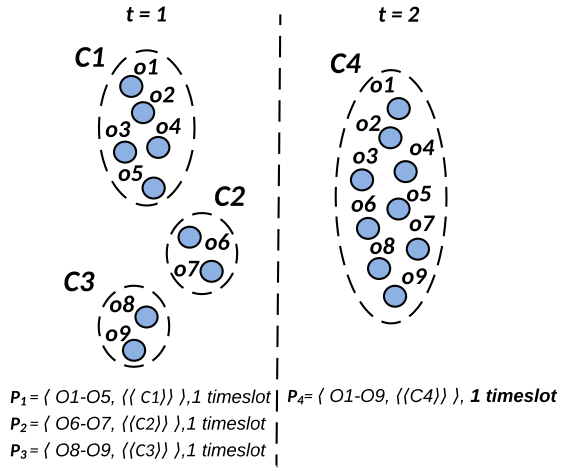


Fig. 5. Opportunistic candidate creation

time-slots of a group of objects. Assuming the size threshold of 2, during the first time-slot, 7 candidates are identified (P_1 – P_7), of which just one of them (P_7) can be extended in the second time-slot.

In the opportunistic approach, we start to identify a group as an LTCP candidate from the first time that all of the members stay in one cluster. Therefore, we ignore the combination of clusters as new candidates, which substantially decrease the candidate set size and time complexity, respectively.

Fig. 5 shows an opportunistic candidate creation in two time-slots. In each time-slot, only single clusters which satisfy the size requirement are added to the candidate list. Therefore, group of $O_G = \{O_1 - O_9\}$ cannot be qualified as an LTCP candidate until the second time-slot which all of its members stay in one cluster.

Lemma 2: If a group could be discovered by the straightforward approach, the opportunistic approach is also able to find it.

Proof: As all the members of an LTCP group should stay together for at least f_C time-slots, postponing the group identification to the first gathering won't lead to the non-recognition of the group.

However, as this approach starts tracking a group from the first gathering of all members, it might loose part of the group's lifetime. Considering the example in Fig. 4, although the candidate P_4 is the union of three candidates in the previous time-slot, the opportunistic approach cannot identify that. To moderate this problem, we perform a simple backward checking to see if there is a match between the new candidate and the nonextended candidates in the previous time-slot. Through the backward checking, three candidates in time-slot 1 would be added to the candidate P_4: $\langle\ O_1 - O9,\ \langle\langle C_1, C_2, C_3\rangle, \langle C_4\rangle\rangle\rangle$. Even with applying backward operation, the opportunistic approach might loose part of the lifetime of the group. However, as groups tend to meet at the early times, according to the observation on real data, this won't affect much on the quality of discovered groups.

Benefiting form Lemma 2 and the backward checking, we propose an opportunistic algorithm, Algorithm 3. Due

Algorithm 3 : Opportunistic Approach

Input: Tr_{DB}, m_G, d_G, f_C
Output: all closed LTCPs
1: $V := \emptyset$; $//set\ of\ current\ LTCP\ candidates$
2: **for** *each time slot* t **do**

$//\ Candidate\ creation\ step$
3: $P'' := \emptyset$;
4: **for** *each cluster* $c \in C$ **do**
if $size(c) \geq m_G$ and c is closed**:**
5: $P_{next} := \langle c.O_G, \langle$c$\rangle\rangle$; $//initialize\ a\ new\ candidate$
6: add P_{next} *to* P'';

7: $V'' :=$ *candidates in* V *which are not extended*;
8: **for** *each candidate* $P' \in P''$ **do**
9: $temp := \emptyset$;
10: **for** *each candidate* $P \in V''$ **do**
11: **if** $P.O_G$ *is a subset of* $P'.O_G$**:**
12: add P to $temp$;
13: **if** $temp.O_G = P'.O_G$**:**
14: $P_{next} := \langle P'.O_G, temp\ o\ P'.TS\rangle$;
15: add P_{next} to V_{next};

16: $V := V_{next}$

to the space limitation, we just mention the modified parts of the algorithm. When adding the new clusters to the candidate set, the algorithm checks if the cluster meets the size requirement and also if there is already a candidate containing the same objects (Lines 3-6). During the backward process, it checks if there is a match between the newly created candidates and the previous ones which couldn't be extended during the current time-slot (Lines 7-15). We can perform the same improvement as the one in smart-and-fast approach to reduce the redundant checking in the backward operation.

Proposition 1: Let n_1 be the size of the object set and n_2 be the total size of the candidate set V. The time complexity of opportunistic candidate creation is up to $O(n_1 * n_2 + n_1)$.

Proof: Suppose there are average m_1 clusters and m_2 candidates, with the average size of s_1 and s_2 for a cluster and a candidate. In the first part, the algorithm needs to check every cluster and the time cost is $O(m_1)$. The backward operation has to check every pair of new candidate and the current candidate, which is not extended. In the worst case, which none of the current candidates are extended, and all of the clusters are added to the new candidate list, the backward step carries out $m_1 * m_2$ comparisons, which every one takes $s_1 * s_2$ time. Since $m_1 * s_1 = n_1, m_2 * s_2 = n_2$, the time complexity of backward step is $O(n_1 * n_2)$ and the total time complexity is $O(m_1 + n_1 * n_2)$. As $m_1 \leq n_1$, the upper bound of complexity of opportunistic candidate creation is $O(n_1 + n_1 * n_2)$.

5 Weakly Continuous Loose Travelling Companion Pattern

In cases with lots of objects in a limited space, i.e., airport, it might that the objects in a group stay in the same cluster

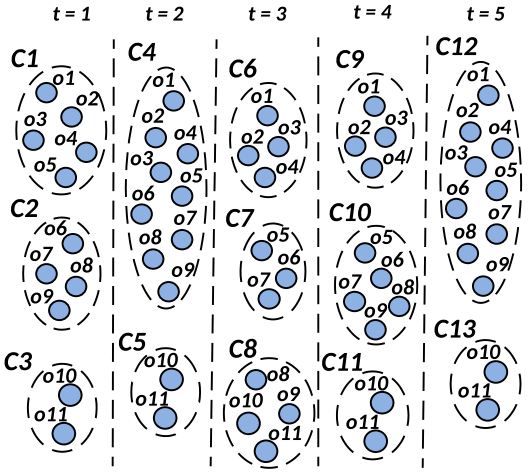


Fig. 6. Example of movement pattern

with other groups for a few time-slots. Strict continuous time constraint in LTCP will prevent the discovery of such group patterns.

Example 7: Fig. 6 shows the movement pattern of two groups of $P_1 = \{O_1 - O_9\}$ and $P_2 = \{O_{10} - O_{11}\}$. In time-slot 3, two members of P_1 stay in a same cluster with P_2. If we set the lifetime threshold $d_G = 4$, the system is not able to discover any patterns, because of the strict time constraint in LTCP. In the case of assuming $d_G = 2$, groups of $\{O_1 - O_9\}$ and $\{O_{10} - O_{11}\}$ are discovered two times with fragmented lifetimes.

Therefore, the rigid continuous time constraints may lead to no discovery of a group or fragmented discovery of a group. For more effective discovery, we propose a **Weakly Continuous Loose Travelling Companion Pattern (WCLTCP)**, which is the extension of LTCP. The difference between WCLTCP and LTCP is the possibility of a time-gap between the cluster-sets.

$Definition\ 3$: Considering the sequence of cluster-sets $TS = \langle cs_{1,a1}, cs_{2,a2}, ..., cs_{n,an} \rangle$ and time-gap threshold l_C, the following condition should be satisfied in a WCLTCP group: $cs_{i+1}.t - cs_i.t \leq l_C$ ($\forall\ i, 1 \leq i < n$).

The main procedure of WCLTCP discovery is the same as the approaches proposed in the previous sections. Only the candidate extension step needs a minor modification. Unlike LTCP, when a candidate cannot be extended, it won't be immediately removed from the candidate set. The system waits to see if the candidate can be extended during a time period of l_C. The general framework of WCLTCP discovery is not affected by such minor changes. As the other steps of the straight forward algorithm, smart-and-fast method, and the opportunistic approach are the same for WCLTCP discovery, we ignore the details here.

6 Experimental Study

In this section, we conduct a series of experiments to evaluate the proposed algorithms.

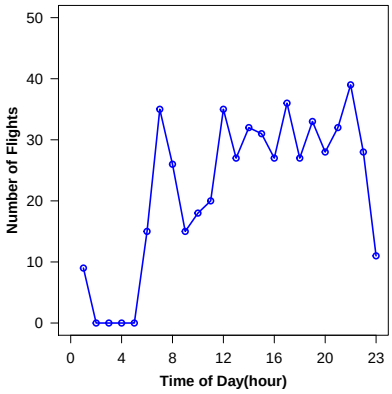


Fig. 7. Distribution of number of flights in a day

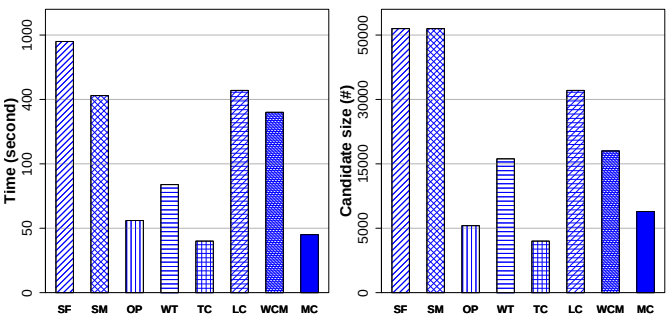
6.1 Experimental Setup

$Dataset$: In order to evaluate the proposed approaches, we use both real and esperimental datasets. The real dataset, D_1, contains trajectories generated by passengers at Guangzhou Baiyun International Airport. As passengers almost spend no more than one day at the airport, we consider a period of a day for collecting trajectories (15 December 2015). Distribution of flight numbers in a day has been shown in Fig. 7. This dataset is comprised of more than 6000 objects, passengers who used the smart trolley during that day, with more than 2 million data records.

In order to collect the location of passengers, we benefit from the new technology, $Charlie$, which provides the greater accuracy than the previous location tracking method at the airport, Radio Frequency Identification/Infrared Identification (RFID/IRID) network. Supplemental information is provided in Acknowledgement. The movement data of Charlie is reported every 10 seconds in the form of $\langle FlightNo, MacAddress, Coordinate, Time, Age, Gender \rangle$. The first four fields are considered to discover the groups of passengers. It should be noted that through the field of MacAddress, we can distinguish Charlies. The remaining fields could be applied to extract more information, like identifying the type of discovered groups, which is out of the scope of this paper.

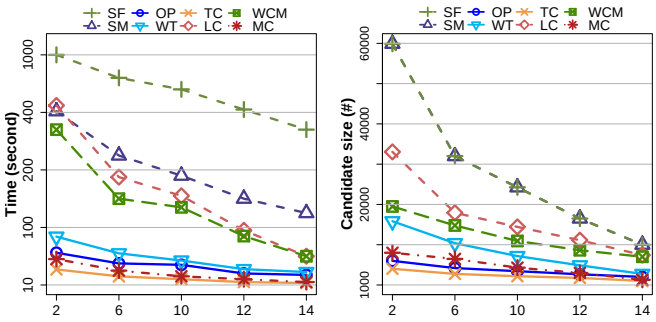
To be able to evaluate the accuracy of the proposed approaches, we conducted an experiment to obtain the ground truth. The experiment took place in Guangzhou Baiyun Airport with 20 participants who used Charlie to browse the airport and report their location. During the experiment, participants were divided into four groups of 2, 4, 6, and 8. Each group followed the predefined routes which were chosen such that the members would split and merge several times. The selected groups spent between 1 to 2 hours browsing the airport. In order to obtain a complete dataset comprising groups with different lengths and lifetime, we also generate a few number of groups according to the real data received from the experiment. The resulted dataset, D_2, is comprised of 14 groups with different size of 2 to 14 and lifetime between 1 to 6 hours which contains more than 100 thousand location points.

$Baselines$: We compare proposed loose travelling companion pattern (LTCP) discovery approaches (StraightForward (SF), Smart-and-Fast (SM) and Opportunistic (OP))



(a) (b)

Fig. 8. Efficiency: (a) time, (b) space



(a) (b)

Fig. 9. Efficiency: (a) time, (b) space, vs m_G

and Weakly Continuous Loose Travelling Companion pattern discovery approach (WT) with the four state-of-the-art baselines: 1) the traveling companion pattern (TC) [12] which captures the groups whose members are close together for certain consecutive time intervals, 2) the loose companion pattern (LC) [13] which is the same as traveling companion pattern with the difference that the gatherings of the whole group can be non-strictly continuous, for certain time intervals, 3) the moving cluster (MC) [14] which allows members to leave the group and new members to join the group at any time, as long as the portion of common members in any two consecutive clusters is not below a given threshold parameter θ. It should be noted that we apply the duration threshold for this pattern, however, in the original version of this algorithm, there is no constraint on the duration, and 4) Weakly consistent group movement pattern (WCM) [18] that allows members to leave the group for a specified time intervals, as long as for each $w - continuous$ clusters there should be at least m_C common members.

It should be noted that TC and LC are implemented based on the Smart-and-Closed algorithm in [12]. We implemented WCLTCP algorithm according to the opportunistic approach.

$Parameter\ setting$: In this paper, we set the parameters according to the observations on the datasets. It should be noted that all of the parameters can vary according to the specific goals. Here, we assume time-slot of 1 minute, and the default setting for both datasets are: $d_G = 20$, $m_G = 2$, $f_C = 10$, $l_C = 10$. For discovery of MC patterns, we set $\theta = 0.5$ and for the WCM patterns, we consider $w = 10$ and $m_C = 2$.

$Environment$: The experiments are conducted on a PC with CPU 1.6 GHz Intel Core i5 and 4.00 GB RAM. The system rans MAC OS X with version 10.11.3. All the algorithms used in these experiments are implemented in $Python$.

6.2 Efficiency

In this subsection we evaluate the efficiency of the algorithms based on the dataset D_1. The clustering phase is the same for all the algorithms, and their runtime costs are omitted for better comparison on the pattern discovery strategy. It should be noted that the parameters of d_G and f_C only have impact on the number of patterns we can discover, and have no effect on the performance (time and space costs) of the algorithms. This is due to the algorithms sequentially passing the time-slots. Therefore, we don't investigate these parameters in our experiments.

We first evaluate the algorithm's time and space costs with the default settings. The size of candidate set (number of objects) is used to measure the space cost. As we can see from Fig. 8, the straightforward approach (SF) has led to a very high time and space cost, which comes from the inefficient candidate extension and candidate creation processes. Through the smart-and-fast approach (SM) we could save more than 50% of the time cost, and the opportunistic approach (OP) significantly improves the performance of LTCP discovery, by upgrading the candidate creation step. Compared to the LTCP (OP algorithm), WCLTCP (WT) results in higher space and time costs, as it allows a time-gap between the cluster-sets. Consequently, the candidates won't be removed immediately from the candidate list, which results in the higher space cost, and accordingly higher time cost.

Travelling companion (TC) reveals the lower cost, in both time and space, compared to the other methods. The reason is that the candidates will be immediately removed from the candidate list, if they are not extended in the current time-slot. On the other hand, LC allows the whole group not to be gathered for a specified time period. Accordingly, even if the candidate is not extended, it will be kept in the candidate list for a specified time period, which leads to the higher space (or time) cost. The reason behind the low efficiency of the WCM is the redundant candidate extension. In this approach, each candidate can be extended to more than one in each time-slot, and as the members can leave the group for a certain period, multiple copy of the same candidate will be generated. Example 4 represents this problem. It should be noted even though MC shows better performance than our approach, it is strongly affected by the value of θ, such that a small decrease in it (e.g., changing it from $\theta = 0.5$ to $\theta = 0.4$) will substantially degrades the performance. This problem has been explained in Section II. Compared to the above baselines, both of the proposed approaches (OP and WT), representing the acceptable performance, significantly better than LC and WCM and close to TC and MC.

In the next step, we evaluate the influence of size thresh-

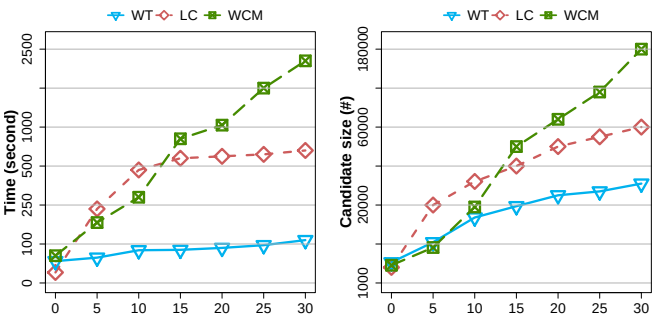


Fig. 10. Efficiency: (a) time, (b) space, vs l_C

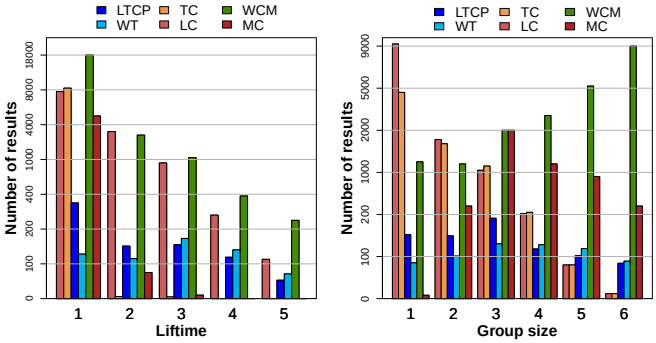


Fig. 11. Effectiveness: (a) lifetime, (b) group size

old m_G on the performance of different approaches, Fig. 9. As m_G increases, the time and space costs of all the algorithms turn to decline. This is generally because the fewer number of clusters (or cluster-sets) can be qualified as the valid candidates. In addition, with increasing the size threshold, the smart cluster-set matching mechanism performs more effective which leads to the lower time cost in two approaches of SM and OP. With increasing the group size threshold, LC shows an abrupt reduction in both running time and the candidate size. The reason is that LC (and also TC) focuses on groups whose members are always together which results in the smaller groups. Therefore, increasing m_G will substantially decrease the size of the candidate set, which subsequently reduces the time cost.

We also investigate the effect of l_C on the performance of the three approaches WCLTCP, LC, and WCM, Fig. 10. In WCLTCP, l_C is the allowed time-gap between the cluster-sets, in LC approach, it determines the time interval that the whole group can not be gathered, and in WCM, it determines the time interval that each member can leave the group. During the previous experiments, we set $l_C = 10$. For this step, we change it from 0 to 30. For $l_C = 0$, WCLTCP is equal to LTCP (OP), and LC and WCM are the same as TC. As l_C increases, space cost of all algorithms rises rapidly. This is because the candidates can stay longer in the candidate list. Consequently, the system needs more time to process the larger candidate list. However, WCM and LC show a stronger increase than WCLTCP. The reason is each candidate can be extended to more than one candidate in each time-slot, and when it comes with the high value of l_C, it leads to the substantial increase in the size of the candidate list. According to Fig. 10, increasing l_C has a greater impact on WCM than LC. This is because of the different definition of l_C in WCM and LC. In LC, l_C is defined as the time intervals that the whole group is allowed not be gathered, while in WCM it determines the number of time intervals that each member can leave the group.

6.3 Effectiveness

In this section, we evaluate the quality of the discovered groups. We conduct our experiments regarding the lifetime and the size of the discovered groups and also the precision of different approaches. The first two experiments are conducted on the first dataset, D_1, and the last one is on the second dataset, D_2. As three proposed approaches for LTCP discovery result the same groups, we just mention LTCP in this part of the experiments.

$Lifetime$ we compare the proposed algorithms and the baselines according to the lifetime of the discovered groups, as it is shown in Fig. 11 (a). The period which groups spend together is divided into 5 categories: (1- less than one hour), (2- one hour to tow hours), (3- two hours to three hours), (4- three hours to four hours), and (5- more than four hours).

As the figure indicates, all of the TC patterns last less than one hour. This is because of putting the strict time continuous requirement which forces the members to stay always connected. Similarly, the groups discovered by MC approach are short-term, they last less than two hours. As it mentioned in Section II, in an MC pattern the portion of common objects in any two consecutive clusters should not be lower than a given threshold, θ. As the lifetime of group increases, the probability of being split into smaller sub-groups also increases. When a group is split into several smaller sub-groups, the overlap between two consecutive clusters decreases. That's the reason why MC approach cannot track the long-term groups. This problem has been explained in Section II. It should be noted that it is strongly affected by the value of θ, such that the lower values lead to the longer term groups. However, at the same time, it significantly degrades the algorithm's performance.

In contrast with TC, LC approach is able to discover the long-term groups, which is because of relaxing the continuous time constraint in LC. However, the reason behind the high number of discovered groups is the partial group discovery of this approach, which will be proved in the next subsection. WCM results in the highest number of groups with different lifetimes. The reason is the loose definition of WCM which results in the wrong and high redundant group discovery. This problem is clearly explained in Section II. Precision results in the next subsection will prove this.

LTCP and WCLTCP, on the other hand, represent the acceptable ability in finding groups with different lifetimes. By the way, as LTCP puts the strict requirement on the continuous cluster-sets, it might discover a single group several times with fragmented lifetimes. This is the reason why LTCP discovers more groups than WCLTCP in some cases. On the other hand, as WCLTCP allows a time-gap between cluster-sets, it is able to find more long-term groups

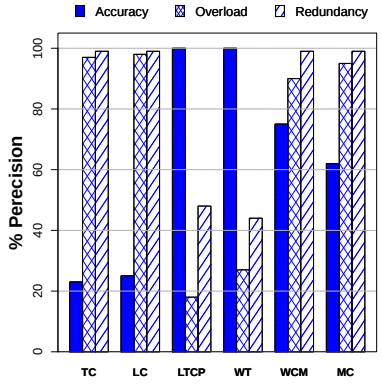


Fig. 12. Precision: $accuracy$, $overload$ and $redundancy$ of different approaches

than the LTCP.

$Groupsize$ For a better comparison, we consider the size of discovered groups as another measure. We define 6 categories for the size of the groups at airport, (1- groups of two), (2- groups of three), (3- groups of four and five), (4- groups of six and seven), (5- groups of eight to ten), (6- groups with more than ten members). Fig. 11 (b) reveals the experiment on the size of the discovered groups in different algorithms.

In total, LC discovers more groups than TC. The reason is the relaxing time constraint in LC leads more candidates be qualified as the group. However, except for the first category (groups of two) which LC detects a greater number, TC and LC algorithms give the same results. It shows that the relaxed time constraint in LC does not make it find more large groups than TC. According to Fig. 11 (b), MC method almost can't discover the groups with size 2. The reason is unlike the TC and LC approaches, which the group is the result of the intersection of the (continuous or with an allowed time-gap) clusters, an MC group is the result of the union of the continuous clusters which satisfy the overlap threshold. This leads to the discovery of large groups. More information can be found in Section II. Contrary to the other approaches, WCM reveals an increasing trend in the discovery of groups with the larger size. It means that this approach discovers more large groups than the small or average groups, which is not consistent with the realistic observations (there is more small and average group in the airport than the large groups). The reason is the loose definition of this approach allows the temporary members to join the group. In addition, WCM results in the redundant group discovery which gets worse when it comes to the larger groups, as it is discussed in Section II.

Unlike the baselines which show the decreasing trend (TC and LC) or increasing trend (WCM), LTCP and WLTCP display the consistent trend regarding different group sizes, which is consistent with the realistic observations. In some cases, LTCP discovers more groups than WCLTCP and in other cases, WCLTCP outperforms it. The reason is explained before, as WCLTCP allows a time-gap between the cluster-sets, it might discover a group that cannot be discovered by the LTCP. On the other hand, LTCP might discover a group several times with fragmented lifetimes, which can be discovered just once with longer lifetime by the WCLTCP.

$Precision$ We consider the information of the groups in dataset D_2 as the ground truth, which the output of the different approaches will be compared with that. The following criteria will be used to evaluate the precision of the different methods, $Accuracy$, $Overload$, and $Redundancy$. We define the criterion of $Accuracy$ as the proportion of the correct discoveries over the ground truth. We also measure the proportion of the wrong discoveries over the retrieved results. Unlike the previous studies which just focused on the incorrect discoveries, here we also measure the redundancy of the algorithms. Considering only the incorrect results as the wrong discoveries measures the $Overload$ criterion. By taking both of the incorrect and redundant discoveries as the wrong results, we measure the $Redundancy$.

Fig. 12 plots the precision of different approaches, according to the default setting. As the figure indicates, TC, LC, LTCP, WCLTCP, WCM, and MC have achieved the accuracy of 23%, 25%, 100%, 100%, 75%, and 62%, respectively. The overload and redundancy of the algorithms are 97%, 98%, 18%, 27%, 90%, 95% and 99%, 99%, 48%, 44%, 99%, 99%, respectively. As the figure shows, both of our proposed approaches, LTCP and WCLTCP, gain the same accuracy. However, LTCP discovers more redundant groups, which is because of the fragmented discovery of a single group. On the other hand, as WCLTCP allows the time-gap between the cluster-sets, it leads to more false positive results, which leads to the higher overload. Because of the relaxed time constraint in LC, it achieves better accuracy than the TC approach. However, both of the TC and LC show the high overload and redundancy in their discoveries. WCM and MC both represent the acceptable ability in discovering the ground truth groups (accuracy), as they do not put the strict continuous time constraint, staying members all the time together. However, these approaches also result in the high number of wrong discoveries (overload and redundancy). The reason is the loose definition of group in these approaches which makes them incapable of discovering the precise groups.

We also study the influence of changing the time threshold d_G, from 10 to 50 time-slots, on the precision of different approaches, Fig. 13. Since all the groups last for at least 60 time-slots in D_2, it won't affect the ground truth. With increasing the d_G, the accuracy of all the approaches decreases. However, LTCP and WCLTCP are still able to discover the high proportion of the right patterns. As it is clear in the figure, accuracy of WCLTCP begins to decrease (in $d_G = 40$) after the LTCP (in $d_G = 30$), which is because of the relaxed time constraint in WCLTCP. The redundancy and overload of the LTCP and WCLTCP also decrease which is mostly because of the lower number of fragmented groups.

On the other hand, LC and TC approaches show the abrupt collapse in accuracy upon increasing the duration threshold, and there is almost no change in redundancy and overload. As we discussed in the two previous subsections, LC and TC approaches discover the high number of long-term small groups (the partial discoveries) and short-term large groups. Accordingly, with increasing the time threshold, redundancy and overload of these two

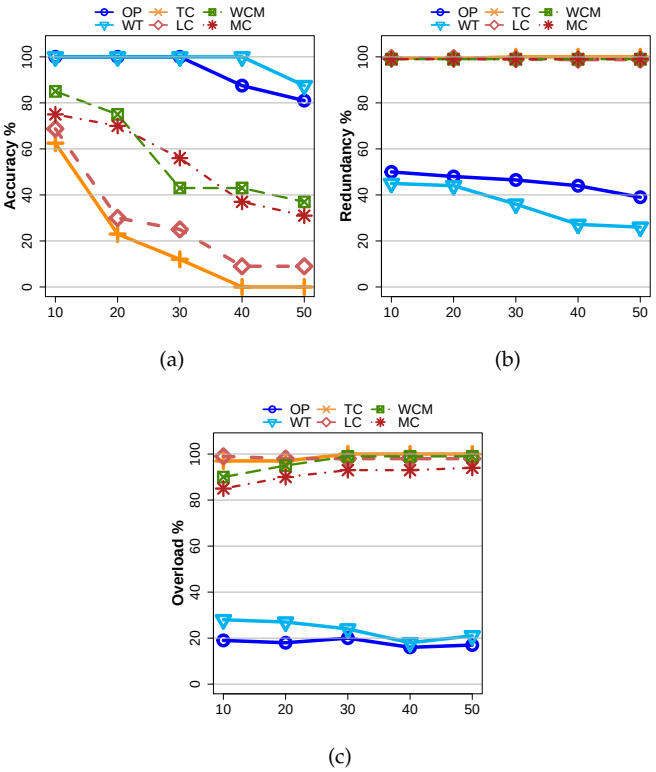


Fig. 13. Precision: (a) accuracy, (b) redundancy, (c) overload, vs d_G

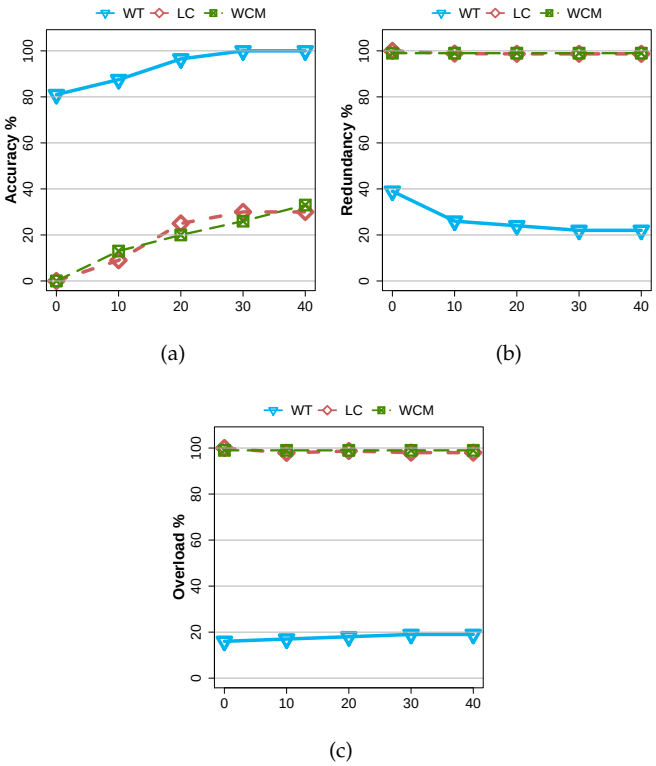


Fig. 14. Precision: (a) accuracy, (b) redundancy, (c) overload, vs l_C

approaches remain unchanged. At the same time, higher time threshold makes them unable to discover the large groups, which leads to the reduction in the accuracy. WCM and MC algorithms also experience a reduction in accuracy, but slower than two other baselines which is because of the non-strict requirement on staying the group members together all the time. On the other side, the overload of these two approaches is increasing, which reveal that the most of the discovered long-term groups by these approaches are wrong. In the end, $d_G = 50$, the accuracy of LTCP, WCLTCP, TC, LC, WCM, and MC falls to 81%, 87.5%, 0%, 12.5%, 37%, and 31%, respectively.

Finally, we perform the experiments with changing the value of the time-gap threshold l_C on the precision of the WCLTCP, WCM, and LC algorithms (the ones who apply this parameter in their model), Fig. 14. In the previous experiments, we consider the default setting. In this part, in order to better illustrate the effects of variation of the l_C, we set $d_G = 50$ and tune l_C from 0 to 30. As shown in Fig. 14, the accuracy of WCLTCP increases with increasing the l_C. It is able to discover all the ground truth groups at $d_G = 50$. The overload of WCLTCP also increases from 16% to 19%, since more groups can be qualified and therefore the number of false-positive results increases. However, the redundancy decreases from 39% to 22%, which is because of the lower number of fragmented discoveries. The LC approach also shows the increase in accuracy result, but it stops at 25%. As the number of wrong discoveries in LC (incorrect or redundant) is too high, increasing the time-gap has no effect on the redundancy and overload. WCM experiences the same result as LC, which reaches the accuracy of 33% at $l_C = 30$. The overload and redundancy of WCM also change as the LC. It should be noted that since database D_2 contains groups even with 2 members, increasing the size threshold m_G leads to losing part of the ground truth. Hence, we omit the precision experiment on this parameter.

As a conclusion, we suggest using the WCLTCP over LTCP in real applications. Even though the WCLTCP results in the higher time and space costs, it achieves the higher accuracy and lower redundant discovery, while keeping the overload at an acceptable level. As the accuracy of the discovered groups by our proposed approaches are not very affected by increasing the duration threshold, it is suggested to set the time threshold relatively high to reduce the incorrect and redundant discoveries. About the time-gap parameter, l_C, we should take this into account that while increasing it improves the accuracy, it leads to the higher overload and more importantly, it degrades the performance (time and space costs) of the algorithms.

6.4 Discussion on Parameter Settings

In the previous section, we conducted various experiments to evaluate the proposed group discovery patterns. As it becomes clear, the parameters value plays an important role in the performance of the algorithms (time and space costs) as well as the quality of the results (precision). Our model is built on four parameters: size threshold (m_G), duration threshold (d_G), frequency threshold (f_C), and time-gap threshold (l_C). The first two parameters are common among all the pattern discovery approaches, except moving cluster which doesn't apply the duration threshold. The

time-gap threshold has been used in WCLTCP, WCM and LC approaches with different applications.

m_G: The value of this parameter is adjusted according to the size of the groups we are searching for, i.e., small groups or big groups. It should be noted that performance of the discovery algorithms is affected by this parameter, such that with increasing the m_G, space cost and time cost would decrease.

d_G: Choosing the proper value for this parameter depends on the application for which groups have been discovered. For example, in a domestic airport passengers spend less time than international airport. Accordingly, the time threshold for the first one should be set with lower values than the latter one. Therefore, it depends on the time that groups are supposed to spend together. The value of d_G also depends on the type of the groups we are searching for. For instance, if we want to discover only the long-term groups, d_G has to be set with higher values. The value of this parameter has no effect on the performance (time and space costs) of the algorithms. This is because the algorithms sequentially passing the time-slots. However, setting d_G with too low values increases the chance of false discoveries, on the other hand, setting it with too high values causes the groups remain undiscovered.

l_C: Different approaches applied this parameter on their model with various ways. In our model, WCLTCP, uses l_C to relax the continuous time constraint in LTCP, third constraint, to let a time-gap between the cluster-sets. The value of this parameter also depends on the application for which groups have been discovered. For instance, for busy hours in an airport, l_C can be set with higher values, as the possibility that members of a group get temporarily involved with other groups increases. However, increasing the l_C, at the same time, degrades the performance of the algorithm by increasing the time and space costs. On the other hand, higher value of l_C increases the accuracy and also increases the chance of false discovery (overload).

f_C: Different from the first category of work, convoy or traveling companion, our approach provides members with freedom to have their own independent movements. On the other hand, unlike the second category of work whose groups might never be totally gathered, we put a constraint on the minimum time-slots that a group has to be gathered, f_C. This parameter is set taking the value of d_G into account, such that it cannot be higher than d_G. Setting f_C with too low values brings it closer to the first category of work which causes the main groups remain undiscovered, while setting it with too high values brings it closer to the second category of work which leads to the false hits.

It should be noted that the previous pattern discovery approaches all have their own specific parameters. For example, evolving convoy applies a window, w, over the continuous clusters ($w - convoy$), and also puts a threshold on the number of common members, m_C, between two continuous $w - convoys$. In weakly consistent movement pattern also there are parameters of w, and m_C which restricts the minimum number of common members between each w continuous clusters. Moving cluster also puts a constraint on the overlap between two continuous clusters, θ.

Fig. 15. Smart trolley used for passenger location tracking at airport

7 Conclusion

In this study, we investigate the problem of discovering the groups of people who travel together from their movement trajectories. Different from earlier proposed patterns, such as convoy, swarm or moving cluster, which aim to identify the general trends from trajectories of animals or vehicles, we aim to identify the precise groups from the movement trajectories of people. The movement of people is rather different from the animal's movement (or vehicles), people might belong to the same main group while they have different movements and contribute in different sub-groups. Accordingly, we introduce a novel group pattern, loose travelling companion pattern, which makes us able to identify the sub-groups in addition to the main group that members contribute. Since the cost of discovering process with the straightforward algorithm could be high because of two major time-consuming operations, we propose smart-and-fast and opportunistic algorithms which improve the candidate extension and candidate creation steps, respectively. The proposed method also is extended to weakly continuous loose travelling companion pattern for more complex scenarios. At last, we evaluate the efficiency and effectiveness of our proposals by conducting the extensive experiments based on the real datasets of passengers moving at the airport. While the proposed approach has shown a good performance, its effectiveness substantially outperforms the baselines in terms of the accuracy, overload, and redundancy. It should be noted that the superiority of our approach becomes more evident when the groups are larger and last longer. As a result, the proposed group pattern is more consistent with the real-life movement pattern of people and can benefit the authorities in understanding the people's behavior and their needs, according to the group(s) that they contribute.

Acknowledgment

Charlie is a smart trolley developed by working with industrial partner Wuxi Chigoo Interactive Technology Co. Ltd in China to help passengers navigate inside the airport and provide personalised services, Fig. 15 [19]. By using this smart trolley, passengers may carry their handbags, receive personalized flight information on the tablet, receive boarding reminders, and enjoy the media and Internet services.

In terms of navigation, a converged positioning technology was implemented to achieve a more accurate position, including invented Radio Frequency Identification/Infrared Identification (RFID/IRID) to combat the signal collision problem of RFID [37], a magnetic fields based positioning for coverage, and the combination of RFID/IRID and magnetic fields to provide accuracy [38]. A the same time, the authors would also like to thank the National Natural Science Foundation of China to sponsor this project in part under grant numbers 61271233 and 61472192, the Scientific and Technological Support Project (Society) of Jiangsu Province under Grant number BE2016776, and also British Council for supporting this PhD placement opportunity to work with Chinese academic staff.

REFERENCES

[1] Z. Li, J. Han, M. Ji, L.-A. Tang, Y. Yu, B. Ding, J.-G. Lee, and R. Kays, "Movemine: Mining moving object data for discovery of animal movement patterns," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 4, p. 37, 2011.

[2] S. Liu, S. Wang, K. Jayarajah, A. Misra, and R. Krishnan, "Todmis: Mining communities from trajectories," in *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*. ACM, 2013, pp. 2109–2118.

[3] Y. Wang, Z. Luo, G. Qin, Y. Zhou, D. Guo, and B. Yan, "Mining common spatial-temporal periodic patterns of animal movement," in *eScience (eScience), 2013 IEEE 9th International Conference on*. IEEE, 2013, pp. 17–26.

[4] C. Loglisci, D. Malerba, and A. N. Papadopoulos, "Mining trajectory data for discovering communities of moving objects." in *EDBT/ICDT Workshops*, 2014, pp. 301–308.

[5] M. Benkert, J. Gudmundsson, F. Hübner, and T. Wolle, "Reporting flock patterns," *Computational Geometry*, vol. 41, no. 3, pp. 111–125, 2008.

[6] M. R. Vieira, P. Bakalov, and V. J. Tsotras, "On-line discovery of flock patterns in spatio-temporal data," in *Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems*. ACM, 2009, pp. 286–295.

[7] H. Jeung, M. L. Yiu, X. Zhou, C. S. Jensen, and H. T. Shen, "Discovery of convoys in trajectory databases," *Proceedings of the VLDB Endowment*, vol. 1, no. 1, pp. 1068–1080, 2008.

[8] H. Jeung, H. T. Shen, and X. Zhou, "Convoy queries in spatio-temporal databases," in *Data Engineering, 2008. ICDE 2008. IEEE 24th International Conference on*. IEEE, 2008, pp. 1457–1459.

[9] Z. Li, B. Ding, J. Han, and R. Kays, "Swarm: Mining relaxed temporal moving object clusters," *Proceedings of the VLDB Endowment*, vol. 3, no. 1-2, pp. 723–734, 2010.

[10] K. Zheng, Y. Zheng, N. J. Yuan, and S. Shang, "On discovery of gathering patterns from trajectories," in *Data Engineering (ICDE), 2013 IEEE 29th International Conference on*. IEEE, 2013, pp. 242–253.

[11] K. Zheng, Y. Zheng, N. J. Yuan, S. Shang, and X. Zhou, "Online discovery of gathering patterns over trajectories," *IEEE Transactions on Knowledge and Data Engineering*, vol. 26, no. 8, pp. 1974–1988, 2014.

[12] L.-A. Tang, Y. Zheng, J. Yuan, J. Han, A. Leung, C.-C. Hung, and W.-C. Peng, "On discovery of traveling companions from streaming trajectories," in *Data Engineering (ICDE), 2012 IEEE 28th International Conference on*. IEEE, 2012, pp. 186–197.

[13] L.-A. Tang, Y. Zheng, J. Yuan, J. Han, A. Leung, W.-C. Peng, and T. L. Porta, "A framework of traveling companion discovery on trajectory data streams," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 5, no. 1, p. 3, 2013.

[14] P. Kalnis, N. Mamoulis, and S. Bakiras, "On discovering moving clusters in spatio-temporal data," in *SSTD*, vol. 3633. Springer, 2005, pp. 364–381.

[15] Y. Li, J. Han, and J. Yang, "Clustering moving objects," in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2004, pp. 617–622.

[16] C. S. Jensen, D. Lin, and B. C. Ooi, "Continuous clustering of moving objects," *IEEE Trans. Knowl. Data Eng.*, vol. 19, no. 9, pp. 1161–1174, 2007.

[17] H. H. Aung and K.-L. Tan, "Discovery of evolving convoys." in *SSDBM*, vol. 6187. Springer, 2010, pp. 196–213.

[18] Y. Wang, Z. Luo, Y. Xiong, D. J. Prosser, S. H. Newman, J. Y. Takekawa, and B. Yan, "Discovering loose group movement patterns from animal trajectories," in *e-Science (e-Science), 2015 IEEE 11th International Conference on*. IEEE, 2015, pp. 196–206.

[19] E. Naserian, X. Wang, X. Xu, and Y. Dong, "Discovery of loose travelling companion patterns from human trajectories," in *High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS), 2016 IEEE 18th International Conference on*. IEEE, 2016, pp. 1238–1245.

[20] Y. Zheng, "Trajectory data mining: an overview," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 6, no. 3, p. 29, 2015.

[21] A. Kharrat, I. S. Popa, K. Zeitouni, and S. Faiz, "Clustering algorithm for network constraint trajectories," in *Headway in Spatial Data Handling*. Springer, 2008, pp. 631–647.

[22] I. V. Cadez, S. Gaffney, and P. Smyth, "A general probabilistic framework for clustering individuals and objects," in *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2000, pp. 140–149.

[23] J.-G. Lee, J. Han, and K.-Y. Whang, "Trajectory clustering: a partition-and-group framework," in *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*. ACM, 2007, pp. 593–604.

[24] Z. Li, J.-G. Lee, X. Li, and J. Han, "Incremental clustering for trajectories," in *International Conference on Database Systems for Advanced Applications*. Springer, 2010, pp. 32–46.

[25] Y. Zheng and X. Xie, "Learning travel recommendations from user-generated gps traces," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 2, no. 1, p. 2, 2011.

[26] A. Monreale, F. Pinelli, R. Trasarti, and F. Giannotti, "Wherenext: a location predictor on trajectory pattern mining," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2009, pp. 637–646.

[27] F. Giannotti, M. Nanni, F. Pinelli, and D. Pedreschi, "Trajectory pattern mining," in *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2007, pp. 330–339.

[28] E. Naserian, X. Wang, X. Xu, Y. Dong, G. N, and H. K, "Integrated discovery of location prediction rules in mobile environment," in *Datacom; 15th Intl Conf on Dependable, Autonomic and Secure Computing, 15th Intl Conf on Pervasive Intelligence and Computing, 3rd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress*. IEEE, 2017, pp. 1017–1024.

[29] H. Cao, N. Mamoulis, and D. W. Cheung, "Discovery of periodic patterns in spatiotemporal sequences," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 4, pp. 453–467, 2007.

[30] J. Yang, W. Wang, and P. S. Yu, "Mining asynchronous periodic patterns in time series data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 15, no. 3, pp. 613–628, 2003.

[31] ——, "Infominer: mining surprising periodic patterns," in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2001, pp. 395–400.

[32] Z. Li, B. Ding, J. Han, R. Kays, and P. Nye, "Mining periodic behaviors for moving objects," in *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2010, pp. 1099–1108.

[33] J. Gudmundsson and M. van Kreveld, "Computing longest duration flocks in trajectory data," in *Proceedings of the 14th annual ACM international symposium on Advances in geographic information systems*. ACM, 2006, pp. 35–42.

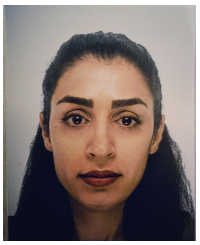
[34] P. Laube and S. Imfeld, "Analyzing relative motion within groups of trackable moving point objects," in *GIScience*, vol. 2. Springer, 2002, pp. 132–144.

[35] Y. Wang, E.-P. Lim, and S.-Y. Hwang, "Efficient mining of group patterns from user movement data," *Data & Knowledge Engineering*, vol. 57, no. 3, pp. 240–282, 2006.

[36] M. Vlachos, G. Kollios, and D. Gunopulos, "Discovering similar multidimensional trajectories," in *Data Engineering, 2002. Proceedings. 18th International Conference on*. IEEE, 2002, pp. 673–684.

[37] W. Lan, H. Zhou, C. Pan, C. Zheng, and T. Chen, "Id identification apparatus," Patent, 2008.

[38] X. Wang, C. Zhang, F. Liu, Y. Dong, and X. Xu, "Exponentially weighted particle filter for simultaneous localization and mapping based on magnetic field measurements," *IEEE Transactions on Instrumentation and Measurement*, 2017.



Elahe Naserian is currently working toward the Ph.D. degree from the School of Engineering and Computing, University of the West of Scotland, United Kingdom. She obtained her master's degree from the University of Tehran, Iran. Her research interests include data mining, machine learning, pattern discovery, and recommender systems.

Xinheng Wang (M'04, SM'14) received his B.E. and M.Sc. degrees in electrical engineering from Xi'an Jiaotong University, Xi'an, China, in 1991 and 1994, and Ph.D. degree in electronics and computer engineering from Brunel University, Uxbridge, UK, in 2001, respectively. He is currently a Professor of Computing with the School of Computing and Engineering, University of West London, London, U.K. He is the investigator/co-investigator of more than 20 research projects sponsored from EU, UK EPSRC, Innovate UK, China NNSFC, and industry. He holds 6 granted and further 12 published invention patents, and has authored or coauthored over 70 referred journal papers. His research has led to a few commercial products in condition monitoring, wireless mesh networks (www.swanmesh.com), and healthcare. His current research interests include indoor positioning, Internet of Things (IoT), and big data analytics for smart airport services, where he has developed the world's first smart trolley with an industry partner.

Xiaolong Xu is a Full Professor in Computing and Software at Jiangsu Key Laboratory of Big Data Security & Intelligent Processing, Nanjing University of Posts and Telecommunications, China. He received his B.Sc. in computer and its applications, M.Sc. in computer software and theories and Ph.D. degree in communications and information systems at Nanjing University of Posts and Telecommunications, Nanjing, China. He is a senior member of China Computer Federation. He has published more than 70 articles as first author or corresponding author in journals, books and conferences in Computer Science and Technology. His main research interests include cloud computing, mobile computing, intelligent agent, and information security.

Yuning Dong (M'07), received his Ph.D. degree from Southeast University in electrical engineering, and his M.Phil. degree in Computer Science from The Queen's University of Belfast (QUB). He is currently a professor with the College of Communications and Information Engineering, Nanjing University of Posts and Telecommunications. at NUPT. He has coauthored over 200 papers in IEEE and other technical journals and referred conference proceedings. He was a British Council postdoctoral fellow at Imperial College London, 1992-93; a visiting scientist at University of Texas, 1993-95; and a research fellow at QUB and the University of Birmingham, 1995-98. His research interests include wireless networking, multimedia communications and network traffic identification.